Inhibited effects in CP-logic[1]

Wannes Meert[a] Joost Vennekens[b]

[a] *Dept. Computer Science, KU Leuven, Belgium*
[b] *Dept. Computer Science – Campus De Nayer, KU Leuven, Belgium*

CP-logic [3, 2] is a Probabilistic Logic Programming language, which is implemented in the Problog system[2]. As a causal, probabilistic modelling tool, it is related to graphical models, such a Bayesian networks. However, its rule-based syntax allows for a more modular and elaboration tolerant representation. Together with its use of logical variables and its capacity for handling cyclic causal relations, this is one of the main benefits of CP-logic. In this paper we present a new language feature for CP-logic where we allow negation in the head of rules to express the inhibition of an effect in a modular manner.

The following example models the spread of an infectious desease from an initially infected group of patients:

$$(\mathit{Inf}(x) : 1) \leftarrow \mathit{InitialInf}(x).$$
$$(\mathit{Inf}(x) : 0.6) \leftarrow \mathit{Contact}(x, y) \wedge \mathit{Inf}(y).$$

The first (deterministic) rule states that all initially infected patients are infected and the second (non-deterministic) rule states that the infection spreads with probability 0.6 from an already infected person to someone he has contact with. Each individual might therefore become infected through numerous different causes. The semantics of CP-logic combines all the associated individual infection probabilities by means of a *noisy-or*. In contrast to Bayesian networks, CP-logic handles causal cycles, so no problems arise when the *Contact* relation is symmetric.

Each rule in CP-logic represents an independent causal mechanism. This gives the language a high degree of elaboration tolerance. For instance, we can extend the above model by a rule to cover the effects of travel to a high-risk area without requiring any changes to the existing model:

$$(\mathit{Inf}(x) : 1) \leftarrow \mathit{InitialInf}(x).$$
$$(\mathit{Inf}(x) : 0.6) \leftarrow \mathit{Contact}(x, y) \wedge \mathit{Inf}(y).$$
$$(\mathit{Inf}(x) : 0.2) \leftarrow \mathit{RiskyTravel}(x).$$

However, CP-logic as originally defined only has this elaboration tolerance when it comes to the addition of new causes, which *raise* the probability of infection. For instance, if we were to discover that certain patients are *less* likely to be infected due to an inherent resistance to the infection, we could not model this without changing our original rules. In this paper, we address this issue by extending CP-logic with *negation in the head*. This new feature does allow us to add the new information without changes to the original rules:

$$(\mathit{Inf}(x) : 1) \leftarrow \mathit{InitialInf}(x).$$
$$(\mathit{Inf}(x) : 0.6) \leftarrow \mathit{Contact}(x, y) \wedge \mathit{Inf}(y).$$
$$(\mathit{Inf}(x) : 0.2) \leftarrow \mathit{RiskyTravel}(x).$$
$$(\neg \mathit{Inf}(x) : 0.3) \leftarrow \mathit{Resistant}(x).$$

[1] This abstract reports on a paper that was been published at the The Seventh European Workshop on Probabilistic Graphical Models (PGM) in 2014 [1].

[2] http://dtai.cs.kuleuven.be/problog

Again, each such rule represents an independent causal mechanism, only now $Inf(x)$ will only hold if it caused by at least one such mechanism *and* if its negation $\neg Inf(x)$ is not caused by any mechanism. For instance, if we consider a particular individual A who was not initially infected, had no risky travel, is resistant and had contact with two infected people, the probability of this individual being infected is:

$$Inf(A) = (0.6 + 0.6 - 0.36) \cdot (1 - 0.3)$$

This feature has been implemented in the Problog system and experiments in the paper show that it incurs a very low overhead.

After publishing the original paper, we learned that researchers from the University of Utrecht independently developed a similar notion in the context of Bayesian networks [4]. In fact, the semantics of the two constructs appears to be identical. For instance, the osteoporosis example [4, example 6.1], can be equivalently modelled in CP-logic as:

$$\begin{aligned}
EnableOsteoblasts &\leftarrow Calcium.\\
StopOsteoclasts &\leftarrow Bispho.\\
(\neg EnableOsteoblasts : 0.5) &\leftarrow Calcium \wedge Bispho.\\
\neg StopOsteoclasts &\leftarrow Calcium \wedge Bispho.\\
(\neg Osteoprosis : 0.85) &\leftarrow StopOsteoclasts.\\
(\neg Osteoprosis : 0.15) &\leftarrow EnableOsteoblasts.
\end{aligned}$$

A more involved examples is the epigastric pains and urinary tract infection example [4, example 6.2]. In this example multiple annihilator variables are introduced, each one for a different combination of the treatments that are administered. Using CP-logic and negation in the head, this example can be expressed as follows (the variables are shortened as in the original example to save space):

$$\begin{array}{lll}
(\neg a_e : 1.00) \leftarrow a \wedge \neg c \wedge p. & (\neg a_e : 1.00) \leftarrow a \wedge c \wedge p. & (\neg e : 0.30) \leftarrow a_e.\\
(\neg p_e : 0.50) \leftarrow a \wedge \neg c \wedge p. & (\neg c_e : 0.50) \leftarrow a \wedge c \wedge p. & (\neg e : 0.95) \leftarrow p_e.\\
(\neg c_e : 0.00) \leftarrow \neg a \wedge c \wedge p. & (\neg p_e : 0.00) \leftarrow a \wedge c \wedge p. & \\
(\neg p_e : 0.57) \leftarrow \neg a \wedge c \wedge p. & a_e \leftarrow a. & (\neg u : 0.00) \leftarrow a_e.\\
(\neg c_e : 0.00) \leftarrow a \wedge c \wedge \neg p. & c_e \leftarrow c. & (\neg u : 0.97) \leftarrow c_e.\\
(\neg a_e : 0.00) \leftarrow a \wedge c \wedge \neg p. & p_e \leftarrow p. & (\neg u : 0.00) \leftarrow p_e.
\end{array}$$

Here, our approach differs slightly from that of [4] by introducing explicit names a_e, p_e and c_e for the latent variables that express whether a, p and c are actually able to exert their effect on e and u. Note that the negative head expressions with a value of zero can be dropped without influencing the model.

Whereas the main motivation for our work was to improve the elaboration tolerance of CP-logic, the Utrecht researchers also report benefits for modeling and knowledge elicitation.

References

[1] Wannes Meert and Joost Vennekens. Inhibited effects in CP-logic. *Probabilistic Graphical Models*, LNCS 8754:350–356, September 2014.

[2] Joost Vennekens, Marc Denecker, and Maurice Bruynooghe. CP-logic: A language of causal probabilistic events and its relation to logic programming. *Theory and Practice of Logic Programming (TPLP)*, 9(3):245–308, 2009.

[3] Joost Vennekens, Sofie Verbaeten, and Maurice Bruynooghe. Logic programs with annotated disjunctions. In *Logic Programming*, pages 431–445. Springer Berlin Heidelberg, 2004.

[4] Steven P.D. Woudenberg, Linda C. van der Gaag, and Carin M.A. Rademaker. An intercausal cancellation model for Bayesian-network engineering. *International Journal of Approximate Reasoning*, 63:32—47, 2015.